User's Guide

To

MMS
Musculoskeletal Modeling In Simulink

By:

Rahman Davoodi

Medical Device Development Facility
Alfred E. Mann Institute for Biomedical Engineering
University of Southern California
http://ami.usc.edu/

August 2002

Table of Contents

1. Introduction 3
2. New in this version 5
3. System requirements and installation 6
3.1. SYSTEM REQUIREMENTS 6
3.2. INSTALLATION 6
4. Building Forward Dynamic Models in SIMM 7
4.1. SKELETAL MODEL 7
4.2. MUSCLE MODEL 7
4.3. EXTERNAL FORCES AND TORQUES 9
5. Using MMS to build Simulink models from SIMM model 10
6. Forward dynamic simulations in Simulink 13
6.1. INPUTS TO THE SIMULINK MODEL 14
Inputs to the muscles 14
Inputs to the prescribed joint axes 14
Inputs to the external point forces 15
Inputs to the external body torques 15
Inputs to the external hinge torques 15
6.2. OUTPUTS OF THE SIMULINK MODEL 16
Outputs for states 16
Outputs for muscles 16
Outputs for sensors 16
Outputs for storage and animation 17
6.3. PARAMETERS OF THE SIMULINK MODEL 18
6.4. RUNNING A SIMULATION AND VIEWING THE RESULTS 21
7. A simple model of arm (an example) 22
7.1. MODEL DESCRIPTION 23
7.2. SKELETAL MODEL 24
7.3. MUSCLE MODEL 26
7.4. CONVERTING SIMM MODEL TO SIMULINK MODEL 29
7.5. SETTING UP AND RUNNING SIMULATION 32
7.6. VIEWING THE RESULTS OF SIMULATION IN SIMM 35
8. MMS License Agreement 36

1. Introduction

Simulink (Mathworks Inc., USA) is a popular software for modeling, simulating, and analyzing dynamical systems. It has many built in blocks and can take advantage of the toolboxes in Matlab. However, it has no utility for development of musculoskeletal models. On the other hand, SIMM (Musculographics Inc., USA) is a popular software for development of sophisticated musculoskeletal models. To perform forward dynamic simulations with SIMM however, the user must write additional C programs to model the muscle activation controllers, feedback sensors, optimization algorithms etc. that is both time consuming and difficult for non-programmers.

Musculoskeletal Modeling in Simulink (MMS) is software that combines the capabilities of SIMM and simulink to interactively design simulink models of the musculoskeletal systems. MMS facilitates the model development process by:

Making the final model available in Simulink where many other utilities are available for data analysis, presentation, control and optimization.
Eliminating the need for writing any user C code by automatically generating the necessary code.
Eliminating restrictions on SIMM model properties such as muscle excitations, external forces and prescribed motions in SIMM and introducing more flexibility while making the model development easier.
Adding a feature to make it possible to add sensors to segments and muscles.
Adding more flexibility in muscle modeling to allow the users to use their own muscle models in addition or in place of the muscle models in SIMM.
Completely integrating the Virtual Muscle software that allows the user to use Virtual Muscle models in combination or in place of the SIMM default muscle models.
Keeping 100% compatibility with SIMM so that the older models developed for SIMM can easily be converted to Simulink models.

In the tradition of not inventing the wheel, we have tried to use the best musculoskeletal modeling software that is currently available and combined it with the most popular dynamic simulation software. Therefore, MMS is a utility that helps to seamlessly integrate these two modeling tools and also adds some very useful features.

In chapter 2, you will find the list of changes and new features in this version of MMS.

In chapter 2, you will find the system requirements and installation procedure for the use of MMS.

Chapter 3 explains the procedure for building a forward dynamic model in SIMM. Because this is covered in SIMM's documentation, I will only explain those aspects that relate to the use of MMS.

Chapter 4 explains the use of MMS to build Simulink models from the SIMM models.

In chapter 5, you will learn how to setup and run the simulink model of your musculoskeletal system.

Finally, chapter 6 uses an example to go through the complete process of building a forward dynamic model and performing simulations.

2. New in this version

The main changes in this version of MMS are as follows:

- A new feature allows the user to attach sensors to the segments or muscles. Sensors of up to 100 can be chosen from eight different artificial and natural sensor types and attached to any point on the segment or muscle.
- In older version of MMS, the user could change the model's initial condition in the final Simulink block's parameter list. This feature is now removed because the new version of SIMM (3.2) allows the user to change these initial values in "forparams.txt" file.
- Initial installation of MMS is now easier because there is no need to manually change some of the SIMM files for compatibility.
- The MMS models can now be run with or without muscles. If you don't provide the muscle file, the muscle related inputs and outputs will not appear in final Simulink model. Such models can be driven by torque motors, external forces and influence of gravity.
- In older version of MMS, the users of Virtual Muscle models had to design them separately and add them to the final Simulink model generated by MMS. This version of MMS has integrated the Virtual Muscle software allowing the user to define the Virtual Muscle parameters inside the SIMM muscle definition file.
- When creating the final Simulink model you are now given a chance to rebuild an existing model or replace it. Rebuilding preserves all the Simulink blocks you may have added to the MMS model.
- This version of MMS is compatible with SIMM 3.2 and Matlab R12/R13

3. System requirements and installation

3.1. System requirements

The following software packages are required for building Simulink models of the musculoskeletal systems.
Windows NT/98/2000
SIMM 3.2 (Musculographics Inc., USA)
Dynamics Pipeline 2.0 (Musculographics Inc., USA)
SD/FAST B.2 (Symbolic Dynamics Inc., USA)
Matlab 6.0 or higher with Simulink (MathWorks Inc., USA)
Microsoft Visual C++ 6.0 (Microsoft Corporation, USA)

3.2. Installation

To use MMS, the software listed in section 3.1 must be installed properly. Except for Matlab and Simulink, the rest of the software in the list are necessary for running forward dynamic simulations with SIMM/Dynamics Pipeline.

To install MMS, Unzip MMS files to a MMS directory (e.g. **C:\MMS**). This is a safe place to store the original MMS files. You will need to copy all of these files to your model directory whenever you create a new model.

To use MMS, copy all the files in **C:\MMS**, to your model directory. For example if you have a SIMM model of arm located in **C:\mymodels\Arm.jnt**, copy all the MMS files to **C:\mymodels.** Keep the original copy in **C:\MMS** for later use with other models.

4. Building Forward Dynamic Models in SIMM

4.1. Skeletal model

The topology of the skeletal model including the segments, joints, degrees of freedom and the physical properties of the segments are defined in the joint file. You must follow the guidelines in SIMM's user manual to write the joint file. With MMS, some model features can be implemented differently that are discussed below:

In SIMM, you can define the joint's axis to be free, or constrained. The constrained joints can be fixed or forced to follow a trajectory. However, once defined, the constraints cannot be modified in runtime without regenerating and compiling the dynamic files.

In MMS however, you can define all joint axis in the joint file to be free. Later when you build the simulink model, you can constrain any joint axis you want. This time, the joint constraints (trajectories) will be provided from within Simulink and can be modified in runtime without rebuilding the model. This provides greater flexibility in application of the constrained motions.

An example of a joint file can be found in chapter 6.

4.2. Muscle model

The muscle geometry, insertion points, line of action and force production models are all defined in muscle file. SIMM allows the user to choose one of their five muscle models or program a new muscle model in C language. The latter may be difficult for some and not justified for those who already have a working muscle model in Simulink.

The MMS however provides the user with two more options for development of Non-SIMM muscle models in Simulink: User defined muscle models and Virtual Muscle models. The procedure for development of these muscle models are explained below:

SIMM muscle models: These are default muscle models that are shipped with SIMM and must be defined according to the guidelines in SIMM's user manual.

Non-SIMM User Defined: For Non-SIMM User Defined muscle models, you still have to follow SIMM's guidelines but:
You only need to provide the parameters that are related to the musculoskeletal geometry such as the attachement points and wrapping objects and the muscle model number that must be **9**.

You need to develop your own muscle model inside Simulink. This model can use the musculotendon length provided by MMS and other information such as the muscle excitation as inputs and must output four variables including muscle excitation, muscle force, normalized fiber length, and normalized fiver velocity to the MMS.

Non-SIMM Virtual Muscle: For Non-SIMM Virtual Muscle, you still have to follow SIMM's guidelines but:

1. You need to add the following to the top of your muscle file to define additional parameters that are required by Virtual Muscle models:

```
begindynamicparameters
mass
optimal_tendon_length
max_musculotendon_length
act_level_when_all_MU_recruited
fiber_type_database
recruitment_type
fractional_pcsa_SS
fractional_pcsa_S
fractional_pcsa_F
number_motor_units_SS
number_motor_units_S
number_motor_units_F
enddynamicparameters
```

2. You need to provide the values of the above parameters in the definition of Virtual Muscle models. You also need to provide optimal_fiber_length and geometric parameters such as attachement points and wrapping objects. Finally you need to use muscle model number **10** to select Virtual Muscles.
3. The Virtual Muscle parameters are those needed by the Virtual Muscles' "Buildmuscles" utility. Selection of these parameters must follow the guidelines of the Virtual Muscle user manual and the following.
    - Currently, you can use SS, S, and F fiber types in "human_fiber_types" or "feline_fiber_types" to build your muscles. You can use "Buildfibertypes" utility to change the parameters of these fiber types but you cannot add fiber types with new names.
    - Unlike the Virtual Muscle, the parameters are given in metric system such as meters and kg.
    - Fiber type database 1 selects "human_fiber_types" and fiber types database 2 selects "feline_fiber_types".
    - Recruitment type 1 selects "Natural" recruitment and recruitment type 2 selects "Intramuscular FES" recruitment.

An example of an input muscle file can be found in chapter 6.

4.3. External forces and torques

External forces and torques are defined in a kinetics file and must be prepared following the SIMM's guidelines. In SIMM's kinetics file, you record the forces and torques you want to apply in each frame. The number of frames, time step between the frames and therefore the total time is fixed. Any change requires the modification of the kinetics file and cannot be done in runtime. So, you cannot change the external forces according to a runtime event or state variables. The forces and torques are all in local reference frame of the segments. In most applications, it is easier to provide the external forces in global reference frame (such as the data from a force plat).

In MMS, you can still use the kinetics file to apply external force and torques if you are comfortable preparing these files or do not want the flexibility of changing them in runtime or when you want to apply the external forces and torques that are expresses in local frames of the segments. However, with MMS, you can now apply your external forces from within Simulink. This could be in addition to the external forces applied in the kinetics file or the only forces applied to the system. In this case, the external forces and torques are expressed in global reference frame and can be modified in runtime as desired.

To apply external forces from Simulink, you must choose the number of forces and torques when you use MMS to build your simulink model. For an example see chapter 6.

5. Using MMS to build Simulink models from SIMM model

If you have prepared the input model files such as joint and muscle files following the SIMM's guidelines and additional guidelines in the previous chapter, you are ready to use MMS to build simulink model for forward dynamic simulation.
First run SIMM, load your SIMM model, select "**save dynamics**" from the file menu, check all the options for "forward simulation" and select the model directory (where you joint and muscle files reside) and click "OK". This will generate all necessary files and save them to a subdirectory called "forward" in your model directory.

To run MMS, open Matlab's command window in your model directory and run MMS script as follows:

>> MMS

You will be asked,

>> Have you changed any of SIMM model files (Y/N)? [N]:

If you have changed the joint file or you are not sure answer yes (Y). This will force regeneration and recompilation of the dynamic files for the modified skeletal topology. If you answer yes, you will also be reminded that you should have run SIMM to regenerate model specific files for the modified model topology.

The default answer is no (N) which means the joint file has not been changed.

Next, you will be asked,

>> Add prescribed joints (Y/N)? [N]:

Here, you can constraint the motion of any joint axis by prescribing its motion. The default answer is no (N), which means you don't have prescribed joints. If you answer yes (Y), you will be presented with the names of the joint axes that are unconstrained (free) and you can choose to prescribe them or leave them unconstrained. For example you may be asked,

>> Prescribe "r_elbow_flexion" (Y/N)? [N]:

Answering yes prescribes "r_elbow_flexion" and answering no (default) leaves it unconstrained.

If you have constrained the motion of a joint axis in the joint file, it won't be available to be prescribed at this stage (you can't prescribe a joint axis twice!). Only the joint axes that are defined as unconstrained in joint file will be available to be prescribed.

Next, you will be asked,

>> Add external point force (Y/N)? [N]:

Answering no (default) means that you don't want to apply point forces. If you answer yes, you will be presented with the names of the segments (all the segments except for ground) where you can choose to apply external point forces or skip. For example, you may be asked,

>> How many point force on "r_ulna"? [0]:

You can enter the number of point forces to be applied to "r_ulna" or "Enter" for no point forces (default).

Next, you will be asked,

>> Add body torque (Y/N)? [N]:

Answer no, if you don't have external body torques. If you answer yes, you will be presented with the names of the segments in the model where you can choose the number of body torques to apply to or skip.

Next, you will be asked,

>> Add hinge torque (Y/N)? [N]:

Answer no, if you don't have external hinge torques. If you answer yes, you will be presented with the names of the joint axes in the model where you can choose the number of hinge torques to apply to or skip.

>> Add sensors (Y/N)? [N]:

Answer no, if you don't have any sensors. If you answer yes, you will be presented with the types of sensors and segments or muscles to attach them to. You can add as many as 100 sensors. Currently, there are eight sensor models including both artificial and natural sensors.

Finally, you will be asked,

>> Enter output simulink model name to build:

Here, you have to enter a name for the simulink model to be built. If there are no Simulink model with similar name, a new model will be created. However if a model with similar name is already open or saved to the current directory, you will be asked whether you want to rebuild or recreat? If you answer recreate, the existing model will be replaced with a new model. All the user added blocks you may have added to the MMS generated Simulink block will be lost. If you answer rebuild (default), the existing model will be rebuilt while keeping the user added Simulink blocks intact. The connection between the MMS and user added blocks however will be disconnected and you may need to reconnect them manually.

The next chapter explains how the created simulink model can be used to perform forward dynamic simulations.

6. Forward dynamic simulations in Simulink

Here, you will learn how to use the simulink model, created by the MMS, to perform forward dynamic simulations.

The MMS builds a simulink model that has a number of inputs and outputs that depend on the joint file and your answers to the questions in the interactive model building process. It has also some adjustable parameters that you can modify. For an example, see chapter 6.

If you double-click on the simulink block to open its subsystem, you will find that all of the inputs and outputs are properly labeled and are all familiar names because they are extracted from your own joint file. So, you will know what to feed to the inputs and how to use the outputs. However, in the following, I will briefly discuss the inputs, outputs and their use.

6.1. Inputs to the Simulink model

Inputs to the muscles

For SIMM muscle models, you will have inputs labeled,

[Muscle name]_act

This input needs to be provided with muscle activation (0-1).

For Non-SIMM User Defined muscle models, you will have inputs labeled

[Muscle name]_act_frc_nfl_nfv

This input needs to be fed with activation, force, normalized fiber length, and normalized fiber velocity. You can use Simulink's library block "Mux" to combine the four signals into one vector (of width four) before connection them to the input. In case you are confused, the activation signal is not really used in any computation. It is directly transferred to the output ports for storage and presentation. Force signal is the output of your Non-SIMM muscle model that is applied to the appropriate points in the skeletal model. Normalized fiber length and velocity (normalized to fiber optimal length) will be used only if you attach a muscle spindle sensor to the muscle. If you don't want to attach muscle spindle to a Non-SIMM User Defined muscle, you can feed these inputs with zero.

For Non-SIMM Virtual Muscle models, you will have inputs labeled

[Muscle name]_act_mtl

This input must be fed with activation (0-1), and musculotendon length. You can use Simulink's library block "Mux" to combine the two signals into one vector (of width two) before connection them to the input. The musculotendon length for all of the muscles are provided by the MMS Simulink block and can be connected to the input of the Virtual Muscle models.

Inputs to the prescribed joint axes

For each prescribed joint axis, there will be three inputs labeled,

[Joint axis name]_pos
[Joint axis name]_vel
[Joint axis name]_acc

You need to feed these inputs with a compatible set of position, velocity and acceleration you want to be prescribed to the joint axis.

Inputs to the external point forces

For each external point force, there will be six inputs labeled,

[Segment name]_pointfrc_px_[Force number]
[Segment name]_pointfrc_py_[Force number]
[Segment name]_pointfrc_px_[Force number]

[Segment name]_pointfrc_vx_[Force number]
[Segment name]_pointfrc_vy_[Force number]
[Segment name]_pointfrc_vz_[Force number]

You need to supply the point of application of the force (in local reference frame of the segment) to the first three inputs and the force vector (in global reference frame) to the last three inputs.

Inputs to the external body torques

For each external body torque, there will be three inputs labeled,

[Segment name]_bodytorque_vx_[Torque number]
[Segment name]_bodytorque_vy_[Torque number]
[Segment name]_bodytorque_vz_[Torque number]

You need to supply the torque vector (in global reference) to these inputs.

Inputs to the external hinge torques

For each external hinge torque, there will be one input labeled,

[Joint axis name]_hingetorque_[Torque number]

You need to feed this input with the hinge torque value. The direction of the torque will be that of the joint axis.

6.2. Outputs of the Simulink model

Outputs for states

This output port outputs the position, velocity and acceleration of all the degrees of freedom in the system. Each output is clearly labeled and can be identified easily. You can connect these outputs to monitor the model outputs, use them as inputs to your feedback controllers etc.

You may see output ports labeled "fixed0, fixed1 …" which you did not expect to see. These are in fact fixed or weld joints that SIMM has created based on your joint file. To implement these fixed joints, SIMM defines a pin joint and then prescribes its motion to force it to stay in its initial position. You can monitor the output ports for fixed joints to verify that these fixed joints don't mode in reality.

Outputs for muscles

For each muscle (SIMM or Non-SIMM), the excitation, force and length are provided in the output with appropriate labels. All these outputs can be used for monitoring the system but musculotendon length will certainly be needed if you have Non-SIMM Virtual Muscle models. In this case, you can connect the appropriate output for the muscle length to your Non-SIMM Virtual Muscle models inputs.

Outputs for sensors

The outputs of the sensors depend on the type of sensor.

Position, velocity, and acceleration sensors output the XYZ components of the position, velocity, and acceleration of a point on a segment or a muscle expressed in the global reference frame.

Accelerometer simulates a 3-axis accelerometer attached to a segment and aligned with its local coordinate frame. Therefore it outputs the XYZ components of the acceleration of the attachment point expressed in the segment's local reference frame.

Gyroscope simulates a 3-axis gyroscope attached to a segment and aligned with its local coordinate frame. Therefore it outputs the XYZ components of the angular velocity of the segment expressed in the segment's local reference frame.

Inclinometer simulates a 3-axis inclinometer attached to a segment and aligned with its local coordinate frame. Therefore it outputs the projections of the gravity vector on the XYZ axes of the segment's local reference frame.

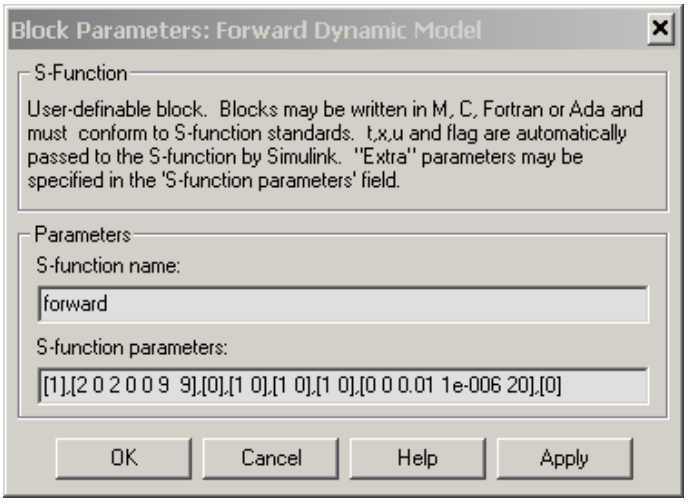
Muscle spindle outputs the primary and secondary afferents firing rates.

Outputs for storage and animation

Here, the skeletal and muscle states are all sent to an output animation file called "motdata.mat". This file is later used to generate SIMM-compatible motion file for viewing the results of simulation in graphic rich SIMM environment.

6.3. Parameters of the simulink model

To see the parameters double-click on the simulink model and then double-click on the block named “forward”. You will see a dialog box similar to the following.


Block Parameters: Forward Dynamic Model
S-Function
User-definable block. Blocks may be written in M, C, Fortran or Ada and must conform to S-function standards. t,x,u and flag are automatically passed to the S-function by Simulink. "Extra" parameters may be specified in the 'S-function parameters' field.
Parameters
S-function name:
forward
S-function parameters:
[1],[2 0 2 0 0 9 9],[0],[1 0],[1 0],[1 0],[0 0 0.01 1e-006 20],[0]
OK
Cancel
Help
Apply

Generally you must not change any of these parameters. But you may need to change few of the parameters that are shown in **bold** in the following.

S-function name is the name of the “dll” file for forward simulation.

S-function parameters contain 8 comma separated parameter vectors. These parameter vectors from left to right are defined below:

1. General Coordinates = [NQ]

Where,
NQ is the number of degrees of freedom. NQ must always be greater than zero otherwise the simulation will stop and error will be reported.

2. Muscles = [NM NSM NNSM NS1 NS2 $MN_{1...}MN_{NM}$]

Where,
NM is the number of muscles in the system.
NSM is the number of SIMM muscle models.
NNSM is the number of Non-SIMM muscle models.
NS1 is the total number of states for SIMM muscle models.
NS2 is the total number of states returned by SIMM for all the muscles.

MN_1 …MN_NM are the muscle model numbers for muscles 1-NM. Muscle model numbers here are those specified in muscle file minus one because the counting starts from 0.
When there are no muscles, Muscles = [0].

3. Prescribed Motion = [NP $PI_{1...}PI_{NP}$]

Where,
NP is the number of prescribed joint axes.
$PI_{1...}PI_{NP}$ are the indices to the prescribed degrees of freedom 1-NP.
When there are no prescribed joint axes, PrescribedMotion = [0].

4. Point Force = [NF $FI_{1...}FI_{NF}$]

Where,
NF is the number of point forces.
$FI_{1...}FI_{NF}$ are the indices to the segments where the point forces 1-NF apply.
When there are no point forces, PointForce = [0].

5. Body Torque = [NB $BI_{1...}BI_{NB}$]

Where,
NB is the number of body torques.
$BI_{1...}BI_{NF}$ are the indices to the segments where the body torques 1-NB apply.
When there are no body torques, BodyTorque = [0].

6. HingeTorque = [NH $HI_{1...}HI_{NH}$]

Where,
NH is the number of hinge torques.
$HI_{1...}HI_{NH}$ are the indices to the joint axes where the hinge torque1-NH apply.
When there are no hinge torques, HingeTorque = [0].

*7. Options = [RUN NSF **FDT TOL BUM**]*

Where,
RUN is the run option and must always be 0.
NSF is equal to 6 times the number of forces plus 3 times the number of torques in the SIMM's kinetic file.
FDT is the sample time between the frames of external force data in the SIMM's kinetic file.

TOL is the tolerance used in assembling the system. Modify if assembly of the system fails.
BUM is the baumgarte constant for constraint stabilization. See SD/FAST manual for further details.

8. Sensors = [NS NSO ST_1 SS_1 SM_1 SP_1 …… ST_{NS} $SS2_{NS}$ $SM2_{NS}$ $SP2_{NS}$*]*

Where,
NS is the number of sensors. When NS=0 none of the following parameters will exist.
NSO is the total number of sensory outputs.
ST_1 …ST_{NS} are the types of sensors 1-NS.
SS_1 …SS_{NS} are the segments where the sensors 1-NS are attached (will be 0 if not used for the sensor type).
SM_1…SM_{NS} are the muscles where the sensors 1-NS are attached (will be 0 if not used for the sensor type).
SP_1 …SP_{NS} are positions (vectors of length 3) of the sensors on the segment or the muscle. If the sensor attach to a segment the vector represents the position of the sensor in segment's local reference frame. When the sensor attach to a muscle, only the middle number will be used as the fraction of the muscle from its origin where the sensor will attach.

6.4. Running a simulation and viewing the results

Once, the simulink block is connected properly to the appropriate inputs and outputs and parameters are adjusted properly, you are ready to run your simulation. Set up your simulation by adjusting the Simulink's simulation parameters (simulation time, integration method etc.) and select start from simulation menu in Simulink.

When the simulation is complete, run Matlab2SIMM script from your Matlab command prompt by typing:

>>Matlab2SIMM

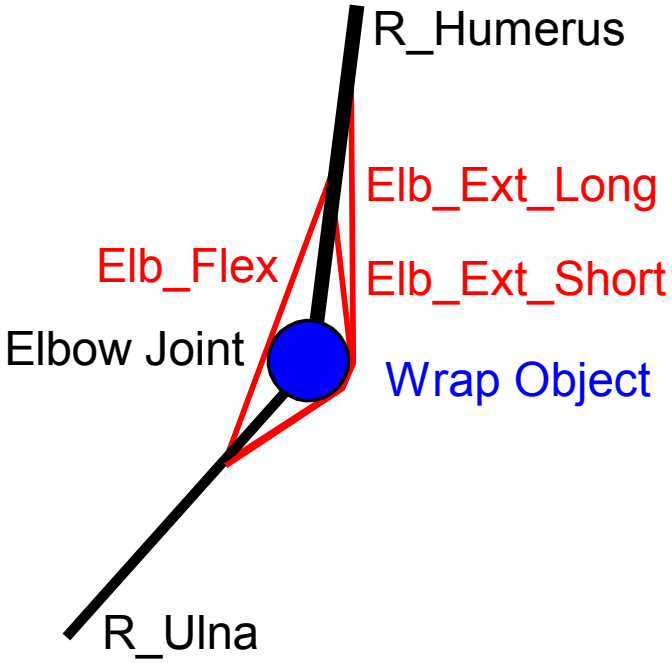
When asked, enter a file name to save the motion data. Once created, you can load the motion file into the SIMM and animate the simulated movement.

7. A simple model of arm (an example)

In the previous chapters you learned how to build a forward dynamic model in SIMM, convert it to a simulink model, setup and run your simulation and view the final results back in SIMM. However, it is always useful to demonstrate the actual use of the procedures by an example. Here, I will use a simple arm model to go through the complete process of model building and simulation.

7.1. Model description

The model is a simple arm model with two segments and one degrees of freedom in elbow. The two segments are right humerus and right ulna that are connected according to the following figure.



A pin joint with one degree of freedom models the elbow joint. Three muscles, Elb_Flex, Elb_Ext_Long, Elb_Ext_Short act on the elbow joint. Two wrap objects are used to keep the elbow muscles in appropriate side of the joint. Note that the model parameters such as mass, inertia are chosen arbitrarily for this example and are not representative of any age group.

7.2. Skeletal model

Skeletal model is defined in the following joint file according to the guidelines in the SIMM's documentation.

/*========= JOINT FILE FOR ARM MODEL ==========*/

```
name ARM
muscle_file arm.msl

/*********************************************/
/*              SEGMENTS                     */
/*********************************************/
beginsegment r_humerus
beginfiles
r_humerus.asc
endfiles
mass 1.000000
masscenter 0.000000 -0.110000 0.000000
inertia 0.0100000000 0.0000000000 0.0000000000
        0.0000000000 0.0010000000 0.0000000000
        0.0000000000 0.0000000000 0.0100000000
axes 0.2
endsegment

beginsegment r_ulna
beginfiles
r_ulna.asc
r_radius.asc
r_hand.asc
endfiles
mass 1.000000
masscenter 0.000000 -0.140000 0.000000
inertia 0.0100000000 0.0000000000 0.0000000000
        0.0000000000 0.0010000000 0.0000000000
        0.0000000000 0.0000000000 0.0100000000
axes 0.2
endsegment

/*********************************************/
/*                JOINTS                     */
/*********************************************/
beginjoint r_elbow
segments r_humerus r_ulna
order t r3 r2 r1
axis1 1.000000 0.000000 0.000000
axis2 0.000000 1.000000 0.000000
axis3 0.000000 0.000000 1.000000
tx  constant 0.014371
ty  constant -0.275795
tz  constant 0.038900
r1  constant 0.000000
r2  constant 0.000000
r3  function f4(r_elbow_flexion)
endjoint

/*********************************************/
/*             GENCOORDS                     */
/*********************************************/
```

```
begingencoord r_elbow_flexion
range 0.00000 150.000000
keys e_key
default_value 0.000000
minrestraint f5
maxrestraint f5
endgencoord

/*********************************************/
/*                FUNCTIONS                  */
/*********************************************/
beginfunction f4
(-360.000000,-360.000000)
( 360.000000, 360.000000)
endfunction

beginfunction f5
(   0.000000,   0.000000)
(  10.000000,  25.000000)
(  20.000000,  50.000000)
(  30.000000,  100.000000)
endfunction

/*********************************************/
/*       MUSCLE WRAP OBJECTS         */
/*********************************************/
beginwrapobject elbowextwrap
wraptype cylinder
segment r_humerus
translation 0.0000 -0.2760 0.0440
radius 0.0300
height 0.0400
quadrant -x
endwrapobject

beginwrapobject elbowflxwrap
wraptype cylinder
segment r_humerus
translation 0.0000 -0.2760 0.0440
radius 0.0300
height 0.0400
quadrant x
endwrapobject
```

/*============== END OF JOINT FILE ==============*/

7.3. Muscle model

The three muscle models are defined in the following muscle file. Elbow flexor uses SIMM muscle model number 1 while elbow extensor is a Non-SIMM muscle model.

/*========== MUSCLE FILE FOR ARM MODEL =======*/

```
/* === PARAMETER NAMES FOR Non-SIMM Virtual Muscles ==*/
begindynamicparameters
mass
optimal_tendon_length
max_musculotendon_length
act_level_when_all_MU_recruited
fiber_type_database
recruitment_type
fractional_pcsa_SS
fractional_pcsa_S
fractional_pcsa_F
number_motor_units_SS
number_motor_units_S
number_motor_units_F
enddynamicparameters


/* ================ DEFAULT MUSCLE ==================*/
beginmuscle defaultmuscle
beginactiveforcelengthcurve
(-5.000000, 0.000000)
( 0.000000, 0.000000)
( 0.401000, 0.000000)
( 0.402000, 0.000000)
( 0.403500, 0.000000)
( 0.527250, 0.226667)
( 0.628750, 0.636667)
( 0.718750, 0.856667)
( 0.861250, 0.950000)
( 1.045000, 0.993333)
( 1.217500, 0.770000)
( 1.438750, 0.246667)
( 1.618750, 0.000000)
( 1.620000, 0.000000)
( 1.621000, 0.000000)
( 2.200000, 0.000000)
( 5.000000, 0.000000)
endactiveforcelengthcurve
beginpassiveforcelengthcurve
(-5.000000, 0.000000)
( 0.998000, 0.000000)
( 0.999000, 0.000000)
( 1.000000, 0.000000)
( 1.100000, 0.035000)
( 1.200000, 0.120000)
( 1.300000, 0.260000)
( 1.400000, 0.550000)
( 1.500000, 1.170000)
( 1.600000, 2.000000)
( 1.601000, 2.000000)
( 1.602000, 2.000000)
( 5.000000, 2.000000)
```

```
endpassiveforcelengthcurve
begintendonforcelengthcurve
(-10.000000, 0.000000)
(-0.002000, 0.000000)
(-0.001000, 0.000000)
( 0.000000, 0.000000)
( 0.001310, 0.010800)
( 0.002810, 0.025700)
( 0.004310, 0.043500)
( 0.005810, 0.065200)
( 0.007310, 0.091500)
( 0.008810, 0.123000)
( 0.010300, 0.161000)
( 0.011800, 0.208000)
( 0.012300, 0.227000)
( 9.200000,345.000000)
( 9.201000,345.000000)
( 9.202000,345.000000)
(20.000000,345.000000)
endtendonforcelengthcurve
beginforcevelocitycurve
(-10.000000, 0.000000)
(-1.000000, 0.000000)
(-0.600000, 0.080000)
(-0.300000, 0.233333)
(-0.100000, 0.540000)
( 0.000000, 1.000000)
( 0.100000, 1.391489)
( 0.300000, 1.593548)
( 0.600000, 1.681481)
( 0.800000, 1.707692)
( 1.000000, 1.724409)
(10.000000, 1.724409)
endforcevelocitycurve
endmuscle


/*==== DEFINE ELBOW FLEXOR AS MUSCLE MODEL 7 ====*/
/*===== MODEL 7 DEFINES A SIMM MUSCLE MODEL =====*/
beginmuscle Elb_Flex
beginpoints
 0.01000 -0.14000  0.02200 segment r_humerus
 0.00200 -0.05000  0.02100 segment r_ulna
endpoints
max_force    100.0
optimal_fiber_length 0.10
tendon_slack_length 0.06
pennation_angle  0.0
max_contraction_velocity 5.00000
wrapobject elbowflxwrap
muscle_model 7
endmuscle

/*== DEFINE ELBOW EXTENSOR 1 AS MUSCLE MODEL 9 ==*/
/*=== MODEL 9 DEFINES NON-SIMM User Defined MUSCLE MODEL ===*/
beginmuscle Elb_Ext_Long
beginpoints
-0.00503 -0.06190  0.01169 segment r_humerus
-0.03000 -0.05000  0.02200 segment r_ulna
endpoints
wrapobject elbowextwrap
muscle_model 9
endmuscle
```

/*== DEFINE ELBOW EXTENSOR 2 AS MUSCLE MODEL 10 ==*/
/*=== MODEL 10 DEFINES NON-SIMM Virtual Muscle MODEL ===*/
beginmuscle Elb_Ext_Short
beginpoints
-0.00255 -0.14472 0.01831 segment r_humerus
-0.03000 -0.05000 0.02200 segment r_ulna
endpoints
wrapobject elbowextwrap
muscle_model 10
optimal_fiber_length 0.1400
mass 0.057890
optimal_tendon_length 0.100000
max_musculotendon_length 0.2800000
act_level_when_all_MU_recruited 0.800000
fiber_type_database 1
recruitment_type 1
fractional_pcsa_SS 0.2
fractional_pcsa_S 0.3
fractional_pcsa_F 0.5
number_motor_units_SS 4
number_motor_units_S 6
number_motor_units_F 10
endmuscle

/* ========== END OF MUSCLE FILE ==========*/

7.4. Converting SIMM model to Simulink model

With the joint and muscle files ready, you can now build the Simulink model for forward simulation.

First you need to load joint and muscle files to the SIMM window and select "Save Dynamics" from the "File" menu to generate the files for forward dynamic simulation. Select the model folder as the simulation folder and check all three boxes for forward simulation. This generates files for forward dynamic simulation in a subdirectory called "forward". See SIMM user manual for more detail.

Next, you should copy MMS files to the model directory if you haven't done so.

Open Matlab in the model directory (where the model files and the MMS files reside) and run MMS. MMS model builder guides you through the complete process of configuring and building the simulink model. Model builder asks a series of questions to interactively configure the model. The questions and answers are given below with their meaning explained.

>> MMS
Explanation: Runs the MMS script in Matlab.

>> Have you changed any of SIMM model files (Y/N)? [N]: y
Explanation: Answer is yes because this is a new model.

>> Have you run SIMM to generate new dynamic files (Y/N)? [Y]:
Explanation: Answer is yes (default) because you should have run SIMM to generate simulation files in the model directory before running MMS.

>> Add prescribed joints (Y/N)? [N]:
Explanation: Answer is no because we have only one degree of freedom and we don't want to prescribe it.

>> Add external point force (Y/N)? [N]: y
Explanation: Answer is yes to apply external point forces.

>> How many point force on "r_ulna"? [0]: 1
Explanation: Answer is 1 to apply 1 external point force on r_ulna. The point of application (in local frame) and the force vector (in global frame) will be provided in runtime.

>> Add body torque (Y/N)? [N]:
Explanation: Answer is no (default) to not apply any body torque.

>> Add hinge torque (Y/N)? [N]:
Explanation: Answer is no (default) to not apply any hinge torque.

>> Add sensors (Y/N)? [N]: y
Explanation: Answer is yes to attach sensors.

[1] - Global position of a point on a segment
[2] - Global velocity of a point on a segment
[3] - Global acceleration of a point on a segment
[4] - Accelerometer aligned with segment axes
[5] - Gyroscope aligned with segment axes
[6] - Inclination with respect to vertical gravity vector
[7] - Global position of a point along a muscles line of action
[8] - Muscle spindle
Select type of sensor: 4
Explanation: Answer is 4 to attach an accelerometer.

[0] - r_ulna
Select a segment to attach the sensor to [0]:
Explanation: Answer is 0 to attach the sensor to r_ulna.

E**nter position of sensor in segment local frame [0 0 0]:**
Explanation: Answer is [0 0 0] to attach the accelerometer to the center of r_ulna's local reference frame.

Add another sensor (Y/N)? [N]: y
Explanation: Answer is yes to attach another sensor.

[1] - Global position of a point on a segment
[2] - Global velocity of a point on a segment
[3] - Global acceleration of a point on a segment
[4] - Accelerometer aligned with segment axes
[5] - Gyroscope aligned with segment axes
[6] - Inclination with respect to vertical gravity vector
[7] - Global position of a point along a muscles line of action
[8] - Muscle spindle
Select type of sensor: 8
Explanation: Answer is 8 to attach a muscle spindle sensor.

[0] - Elb_Flex
[1] - Elb_Ext_Long
[2] - Elb_Ext_Short
Select the muscle to attach the sensor to [0]: 2
Explanation: Answer is 2 to attach a muscle spindle to Elb_Ext_Short.

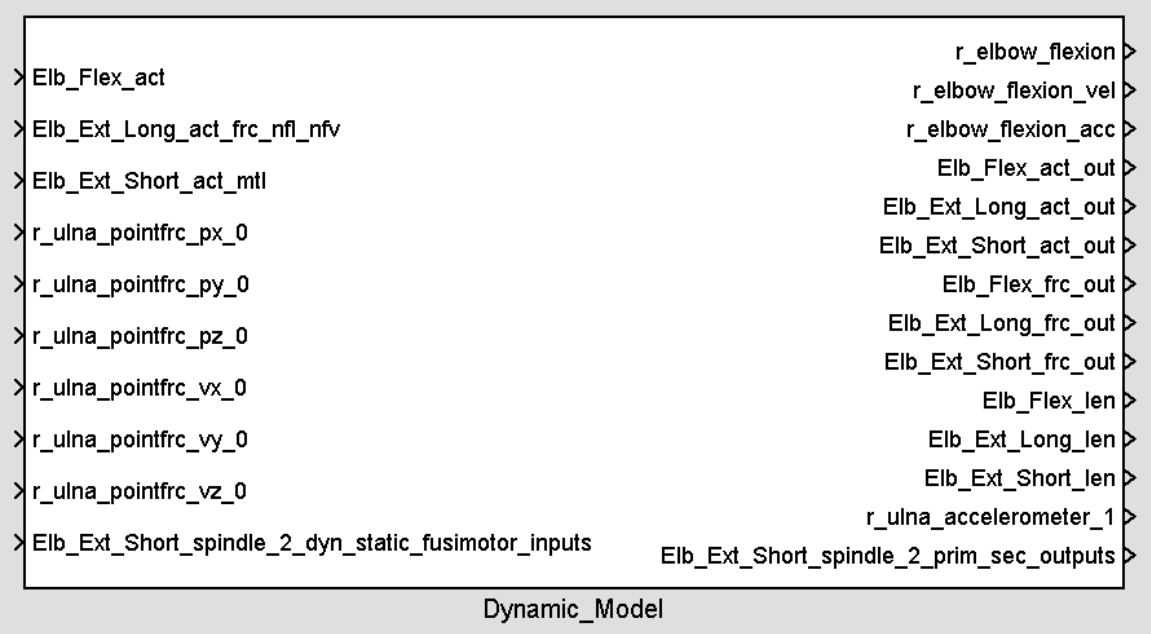
Add another sensor (Y/N)? [N]:
Explanation: Answer is no to not attach any more sensors.

>> Enter output simulink model name to build: arm
Explanation: Answer is "arm" to name the simulink model as "arm.mdl".

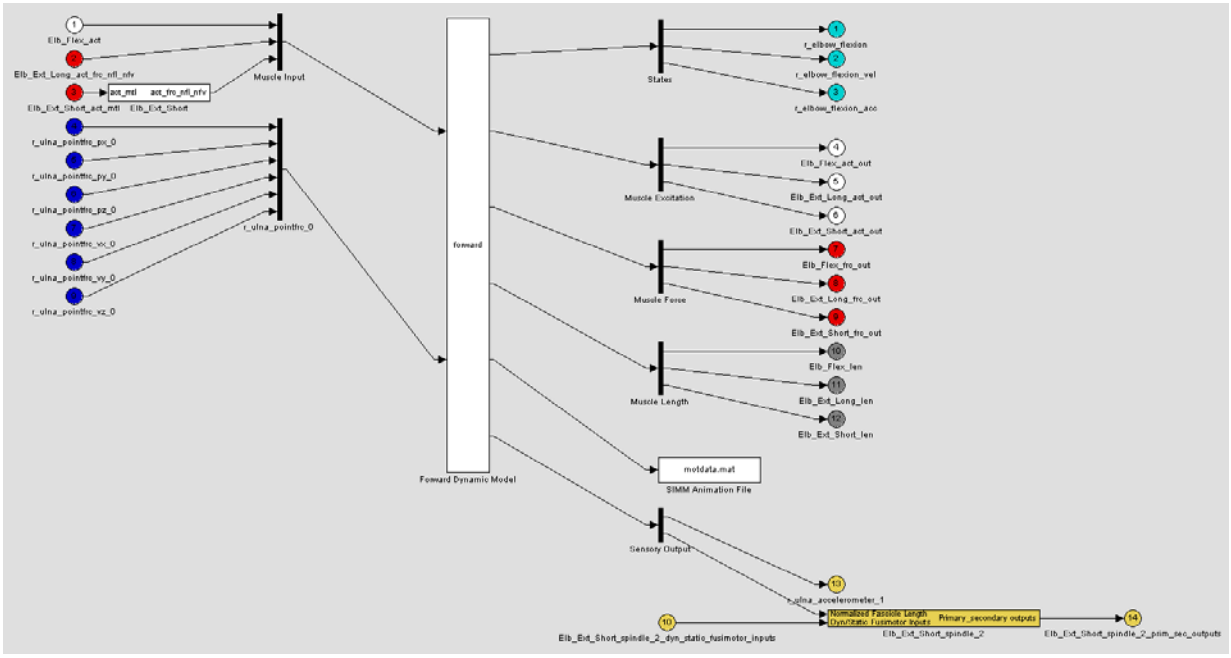
The "arm.mdl" is the simulink model for forward simulation. You need to connect the inputs and outputs before running the simulation. This is explained in the next section.

7.5. Setting up and running simulation

The simulink model, “arm.mdl” which was generated in the last section looks like the following. It has all the necessary inputs and outputs with appropriate labels.



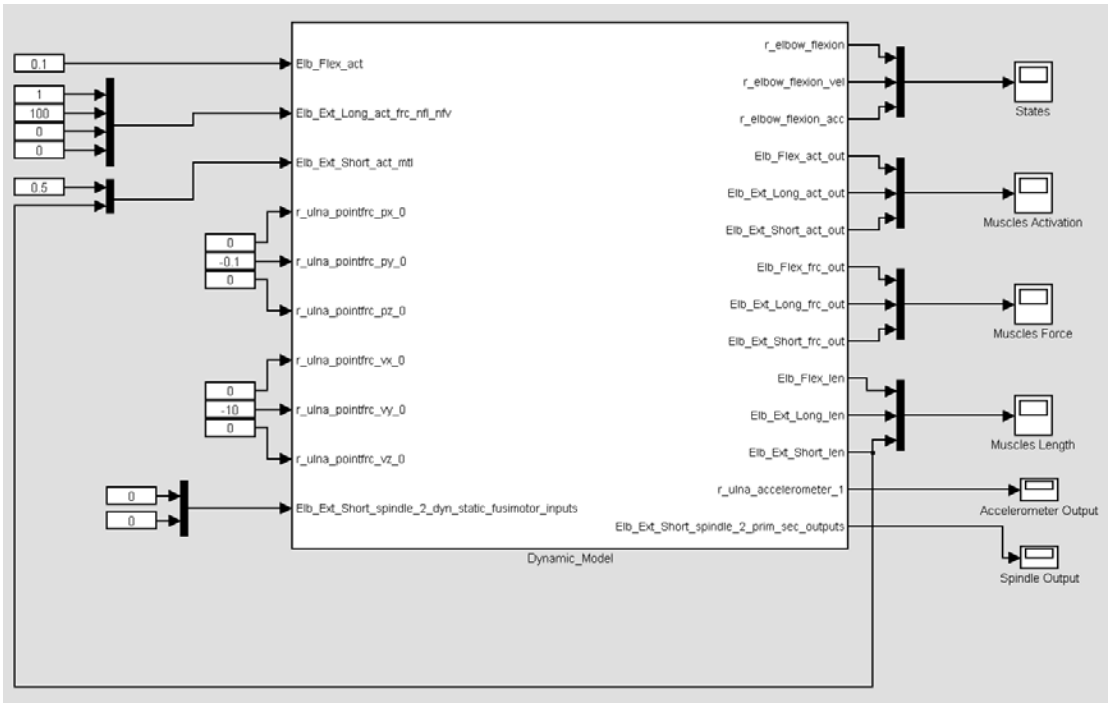
Double-clicking on the Simulink block reveals its subsystem as follows:



As an example of how an MMS model may be connected to other blocks we make the following connections to prepare the model for simulation:

- Elb_Flex_act is a default SIMM muscle (as indicated in input muscle file) and needs only activation as its input, which is connected to a constant input of 0.1.
- Elb_Ext_Long is a Non-SIMM User Defined muscle and needs four inputs. These inputs must come from a muscle model developed by the user. Because we don't have one here, we provide arbitrary values of 1 as activation, 100 N as muscle force, and 0 for fiber length and velocity.
- Elb_Ext_Short is a Non-SIMM Virtual Muscle and need two inputs: activation and musculotendon length. Activation input is connected to a constant source of 0.5. The musculutendon input comes from the output of the MMS block.
- Connect px, py and pz of the external force on r_ulna to 0, -0.1 and 0 meaning that the force will be applied to the hand. Connect the vx, vy and vz components to 0, -10 and 0 meaning that an external force of 10 N will be applied in the direction of -Y of the global reference frame (downward direction, as if an object weighing about 1 kg is carried by the hand).
- Connect the output ports to the Scopes to view their time course during the simulation.

The resulting system will look like the following, which can be simulated by selecting "Start" from Simulink's "Simulation" menu.

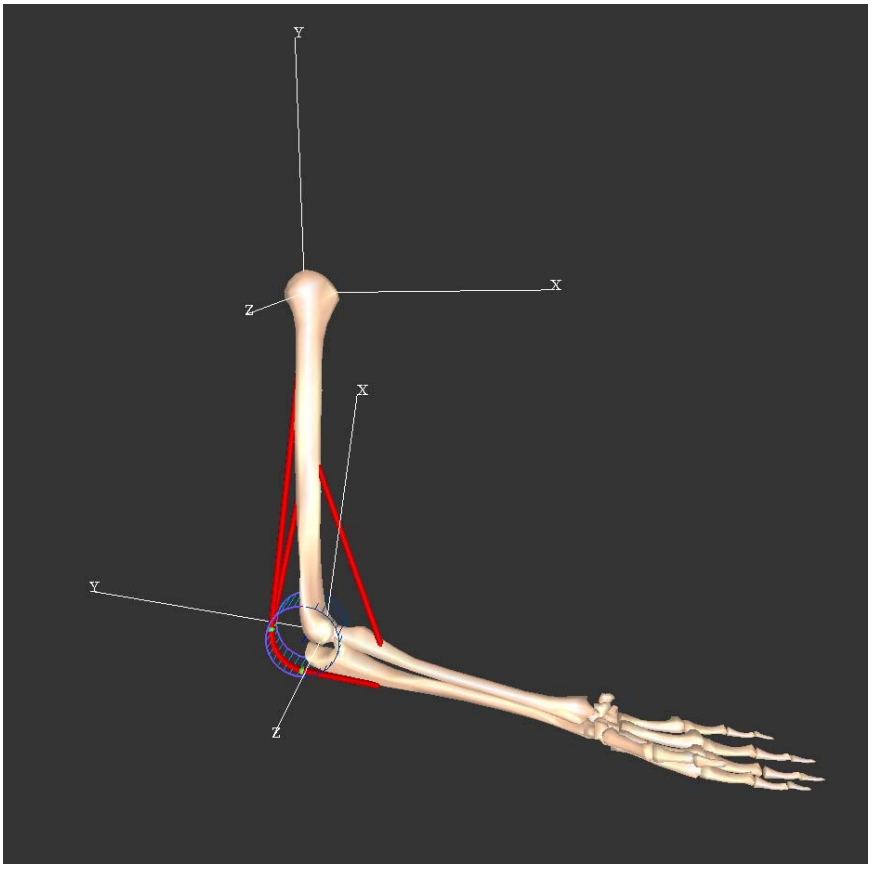


7.6. Viewing the results of simulation in SIMM

Once the simulation run is complete, run the "Matlab2SIMM" script in Matlab by typing:

>>Matlab2SIMM

When asked choose the name of the motion file to create. Then open the model and motion file in SIMM and view the animation of the simulated movement.



8. MMS License Agreement

Alfred E. Mann Institute (AMI) at the University of Southern California

This agreement is being entered into to permit exchange of proprietary information for the pursuit of scientific research and training. It is intended to protect software under development pending completion of its development and a decision about its proper dissemination and/or commercialization.
If you use the Musculoskeletal Modeling in Simulink (MMS) software, you agree to be bound by the terms and conditions of this agreement, and this will be a legally binding agreement between you and Alfred E. Mann Institute.

AMI hereby grants to you a personal, non-transferable, and non-exclusive copy of MMS free of charge to possess, duplicate, use, and execute, solely under the terms of this agreement. You may not sublicense, distribute, hypothecate, lease, loan, or otherwise convey the MMS or any portion thereof to anyone, and under no circumstance may you use or allow the use of MMS in any manner other than as expressly set forth above. You agree that you will not reverse assemble, decompile, or disassemble or otherwise reverse engineer any portion of the MMS.

You agree and acknowledge that AMI transfers no ownership interest in the MMS, in the intellectual property to you under this Agreement.

AMI makes no warranties, expressed or implied, and neither AMI nor any individuals or entities associated with it shall be liable, under any circumstances, for any special consequential, exemplary, or punitive damages, occurring out of or in connection with the use or performance of the materials covered herein.

I, _____, agree with the terms and conditions of this agreement.

Signature: _____

Date: _____

Please mail the completed form to: *Rahman Davoodi, 1042 West 36th Place, Room B12, University of Southern California, Los Angeles, CA 90089-1112* ***OR fax it to:*** *(213) 821-1120*